3ETS+RD-LSTM: A New Hybrid Model for Electrical Energy Consumption Forecasting*

Grzegorz Dudek (✉)[1][0000−0002−2285−0327], Paweł Pełka[1][0000−0002−2609−811X], and Slawek Smyl[2][0000−0003−2548−6695]

[1] Electrical Engineering Faculty, Czestochowa University of Technology, Czestochowa, Poland
{dudek,p.pelka}@el.pcz.czest.pl
[2] Uber Technologies, 555 Market St, San Francisco CA 94104, USA
slaweks@hotmail.co.uk

Abstract. This work presents an extended hybrid and hierarchical deep learning model for electrical energy consumption forecasting. The model combines initial time series (TS) decomposition, exponential smoothing (ETS) for forecasting trend and dispersion components, ETS for deseasonalization, advanced long short-term memory (LSTM), and ensembling. Multi-layer LSTM is equipped with dilated recurrent skip connections and a spatial shortcut path from lower layers to allow the model to better capture long-term seasonal relationships and ensure more efficient training. Deseasonalization and LSTM are combined in a simultaneous learning process using stochastic gradient descent (SGD) which leads to learning TS representations and mapping at the same time. To deal with a forecast bias, an asymmetric pinball loss function was applied. Three-level ensembling provides a powerful regularization reducing the model variance. A simulation study performed on the monthly electricity demand TS for 35 European countries demonstrates a high performance of the proposed model. It generates more accurate forecasts than its predecessor (ETS+RD-LSTM [1]), statistical models such as ARIMA and ETS as well as state-of-the-art models based on machine learning (ML).

Keywords: Exponential smoothing · Long short-term memory · Mid-term load forecasting.

1 Introduction

The power system load is a nonlinear and nonstationary process that can change rapidly due to many factors such as macroeconomic variations, weather, electricity prices, consumer types and habits, etc. Therefore, electricity demand forecasting, which is essential for the power system operation and planning, is a big challenge. In this study we consider mid-term electrical load forecasting (MTLF) focusing on monthly electricity demand forecasting over 12 months horizon.

* The project financed under the program of the Polish Minister of Science and Higher Education titled "Regional Initiative of Excellence", 2019-2022. Project no. 020/RID/2018/19, the amount of financing 12,000,000.00 PLN.

MTLF methods can be roughly classified into statistical/econometrics methods or ML methods [2]. The former include ARIMA, ETS and linear regression (LR). ARIMA and ETS can deal with seasonal TS but LR requires additional operations such as decomposition or extension of the model with periodic components [3]. Limited adaptability of the statistical MTLF models and problems with nonlinear relationship modeling have increased researchers' interest in ML and AI tools [4]. Of these, neural networks (NNs) are the most popular because of their attractive features including learning capabilities, universal approximation property, nonlinear modeling and massive parallelism. Some examples of using NNs for MLTF are: [5] where NN uses historical loads and weather variables to predict monthly demand and is trained by heuristic algorithms to improve performance, [6] where Kohonen NN is used, [7] where NNs are supported by fuzzy logic, [8] where generalized regression NN is used, [9] where weighted evolving fuzzy NN is used, and [10] where NNs, LR and AdaBoost are combined.

Recent trends in ML such as deep recurrent NNs (RNNs), are very attractive for TS forecasting [11]. RNNs are able to exhibit temporal dynamic behavior using their internal state to process sequences of inputs. Recent works have reported that RNNs, such as the LSTM, provide high accuracy in forecasting and outperform most of the traditional statistical and ML methods [12]. Some application examples of LSTMs to load forecasting can be found in [13–15].

In [1] we proposed a hybrid residual dilated LSTM and ETS model (ETS+RD-LSTM) for MTLF. This model was based on the winning submission to the M4 forecasting competition 2018 [16], developed by Slawek Smyl [17]. A simulation study confirmed the high performance of the model and its competitiveness with classical models such as ARIMA and ETS as well as state-of-the-art ML models. In this work we extend ETS+RD-LSTM by introducing initial TS normalization, i.e. detrending and unifying the variance. This method of TS preprocessing we used in our previous works achieving very good results [18]. Recently we used it for LSTM model obtaining a 15% reduction in error [19]. We expect that the TS initial normalization, which simplifies the relationship between input and output data, allows ETS+RD-LSTM to improve its performance.

2 Forecasting Model

The proposed model is a modified version of ETS+RD-LSTM which we described in [1]. We extend ETS+RD-LSTM by introducing initial TS normalization. A normalization procedure removes a trend and unifies variance of the TS. The normalized TS exhibit yearly patterns which are further removed using deseasonalization as an integral part of ETS+RD-LSTM. The normalized and deseasonalized TS are forecasted using RD-LSTM. Then the forecasts are reseasonalized using seasonal components extracted by ETS. To reduce the model variance, we use ensembling at three levels which aggregates individual forecasts. The resulting aggregated forecasts are finally denormalized. For denormalization, the forecasts of the mean yearly demand and yearly dispersion are needed. They are produced by additional two ETS modules.

2.1 Architecture and Features

An architecture of the proposed forecasting system is shown in Fig. 3. The system components are as follows (symbols in italics denote the sets of TS or forecasts):

- Normalization – each original monthly electricity demand TS is normalized. This procedure removes a trend from the TS and unifies its variance. Normalization module loads a set of TS (Z), calculates the series of yearly mean demands ($\overline{Z}$) and yearly dispersions (Σ) for each TS, and determines normalized series (Y).
- ETS – exponential smoothing modules for forecasting the yearly mean demands and their dispersions. These values are necessary for denormalization.
- Deseasonalization – each normalized TS is deseasonalized. This procedure extracts the seasonal components, S, individually for each series using ETS (ETSd module), and determines deseasonalized TS, X.
- RD-LSTM – residual dilated LSTM for forecasting the normalized and deseasonalized TS, X.
- Reseasonalization – each TS forecast produced by RD-LSTM is reseasonalized using inverse operations to deseasonalization.
- Ensembling – the reseasonalized forecasts produced by RD-LSTM are averaged. The ensembling module receives the sets of individual forecasts, $\hat{Y}^r_k$, and returns an aggregated forecast for each TS, $\hat{Y}_{\text{avg}}$.
- Denormalization – the averaged forecasts $\hat{Y}_{\text{avg}}$ are denormalized using forecasted values of the yearly means, $\hat{\overline{Z}}$, and dispersions, $\hat{\Sigma}$.

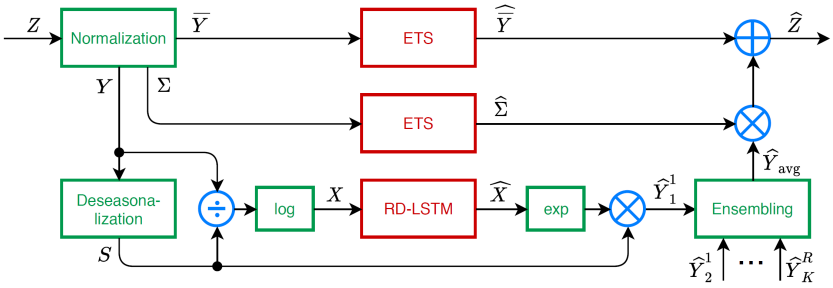

Fig. 1. The proposed forecasting system architecture

The proposed system has a hybrid and hierarchical structure. It combines statistical modeling (ETS), advanced ML (RD-LSTM), and ensembling. ETS is used as a forecasting model for yearly means, $\overline{Z}$, and dispersions, Σ, as well as for extraction of seasonal components (ETSd). The preprocessed TS, without trend and seasonal variations, are forecasted using RD-LSTM. Details of data preprocessing and flow are described in subsection 2.2.

The TS are exploited in a hierarchical manner, meaning that both local and global components are utilized in order to extract and combine information at

either a series or a dataset level, thus enhancing the forecasting accuracy. The global features are learned by RD-LSTM across many TS (cross-learning). The specific features of each individual TS, such as trend, variance, and seasonality, are extracted by normalization and ETSd modules. Thus, each series has a partially unique and partially shared model.

The strength of RD-LSTM, which revealed in M4 competition, is cross-learning, i.e., using many series to train a single model. This is unlike standard statistical TS algorithms, where a separate model is developed for each series. Another important ingredient in the success of the proposed method precursor in the M4 competition was the on-the-fly preprocessing that was an inherent part of the training process. Crucially, the parameters of this preprocessing (in the proposed model these are twelve initial seasonal components and smoothing coefficient β, see subsection 2.2) were being updated by the same overall optimization procedure (SGD) as weights of RD-LSTM, with the overarching goal of minimizing forecasting errors. This enables the model to simultaneous optimization of data representation, i.e. searching for the most suitable representations of input and output data for RD-LSTM, and forecasting performance.

ETSd is used as the preprocessing tool. It extracts a seasonal component which is used for deseasonalization of the normalized TS. ETSd was inspired by the Holt-Winters multiplicative seasonal model. However, it has been simplified by removing trend and level components (see subsection 2.2). This is because the input TS are normalized, i.e. they have no trend and their level is one. ETSd is optimized simultaneously with RD-LSTM using pinball loss function [17]:

$$L_t = \begin{cases} (x_t - \hat{x}_t)\tau & \text{if } x_t \geq \hat{x}_t \\ (\hat{x}_t - x_t)(1-\tau) & \text{if } \hat{x}_t > x_t \end{cases} \tag{1}$$

where x_t and $\hat{x}_t$ are the actual and forecasted values, respectively, and $\tau \in (0, 1)$ is a parameter controlling the loss function asymmetry.

When $\tau = 0.5$ the loss function is symmetrical and penalizes positive and negative deviations equally. When the model tends to have a positive or negative bias, we can reduce the bias by introducing τ smaller or larger than 0.5, respectively. Thus, the asymmetric pinball loss function, penalizing positive and negative deviations differently, allows the method to deal with bias.

ETS for forecasting the yearly mean demands and their dispersions are defined as innovations state space models [20]. They combine the seasonal, trend and error components in different ways (additively or multiplicatively). For each TS the optimal ETS model is selected using Akaike information criterion (AIC).

Ensembling is used for reduction the model variance related to the stochastic nature of SGD, and also related to data and parameter uncertainty. Ensembling is seen as a much more powerful regularization technique than more popular alternatives, e.g. dropout or L2-norm penalty [21]. In our case, ensembling combines individual forecasts at three levels: stage of training level, data subset level and model level. At the stage of training level, the forecasts produced by L most recent training epochs are averaged. This can reduce the effect of stochastic searching, i.e. calming down the noisy SGD optimization process. At the data

subset level, we use K models which learn on the subsets of the training set, Ψ_1, Ψ_2, ..., Ψ_K. Each k-th model produces forecasts for TS included in its own training subset Ψ_k. Then the forecasts produced by the pool of K models are averaged individually for each TS. The third level of ensembling simply averages the forecasts for each TS generated in R independent runs of a pool of K models. In each run, the training subsets Ψ_k are created anew (see [1] for details).

2.2 Time Series Processing

A monthly electricity demand TS exhibits a trend, yearly seasonality and random component (see Fig. 2(a)). To simplify the forecasting problem, the TS is preprocessed as follows. Let $\{z_t\}_{t=1}^{N}$ be a monthly electricity demand TS starting from January and ending in December. This TS is divided into yearly subsequences $\{z_t^i\}_{t=12(i-1)+1}^{12(i-1)+12}, i = 1, ..., N/12$. Each i-th subsequence is expressed by a vector $\mathbf{z}_i = [z_{i,1} z_{i,2} \dots z_{i,12}]^T$. The normalized version of $\mathbf{z}_i$, $\mathbf{y}_i = [y_{i,1} y_{i,2} \dots y_{i,12}]^T$, is determined as follows:

$$y_{i,j} = \frac{z_{i,j} - \overline{z}_i}{\sigma_i} + 1 \tag{2}$$

where $j = 1, ..., 12$, $\overline{z}_i$ is a mean of subsequence $\{z_t^i\}$, and $\sigma_i = \sqrt{\sum_{j=1}^{12}(z_{i,j} - \overline{z}_i)^2}$ is a measure of its dispersion.

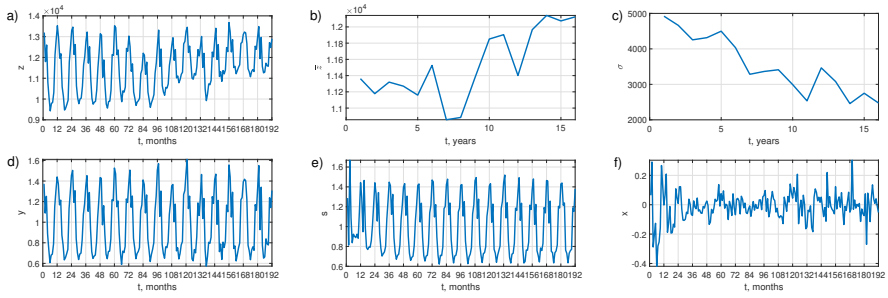

Fig. 2. TS preprocessing: (a) original TS $\{z_t\}$, (b) yearly mean demand TS $\{\overline{z}_t\}$, (c) yearly dispersion TS $\{\sigma_t\}$, (d) normalized TS $\{y_t\}$, (e) seasonal component TS $\{s_t\}$, and (f) normalized and deseasonalized TS $\{x_t\}$

Note that yearly subsequences $\{z_t^i\}$ have different means and dispersions (see Fig. 2(a)). After normalization they are unified, i.e. all yearly subsequences have an average of one, the same variance and also a unity of length. They carry information about the shapes of the yearly sequences. Now we create a new TS composed of normalized subsequences representing successive yearly periods: $\{y_t\} = \{\mathbf{y}_i^T\}_{i=1}^{N/12} = \{y_{1,1}, y_{1,2}, ..., y_{N/12,12}\}$. This TS is shown in Fig. 2(d). Note its regular character and stationarity.

The TS of the mean yearly demand, $\{\overline{z}_i\}_{i=1}^{N/12}$, and yearly dispersion, $\{\sigma_i\}_{i=1}^{N/12}$, are shown in Fig. 2(b) and (c), respectively. They are forecasted by ETS one step ahead (for the next year) and used for denormalization.

The normalized TS, $\{y_t\}$, is further deseasonalized. To do so, we use a simplified Holt-Winters multiplicative seasonal model with only one component:

$$s_{t+12} = \beta y_t + (1-\beta)s_t \tag{3}$$

where s_t is the seasonal component at timepoint t and $\beta \in [0,1]$ is a smoothing coefficient.

The seasonal component is shown in Fig. 2(e). It is used for deseasonalization during the on-the-fly preprocessing. The TS $\{y_t\}$ is deseasonalized in each training epoch using the updated values of seasonal components. These updated values are calculated from (3), where parameters, 12 initial seasonal components and β, are increasingly fine tuned for each TS in each epoch by SGD.

The TS is deseasonalized using rolling windows: input and output ones. The input window contains twelve consecutive elements of the TS which after deseasonalization will be the RD-LSTM inputs. The corresponding output window contains the next twelve consecutive elements, which after deseasonalization will be the RD-LSTM outputs. The TS fragments inside both windows are deseasonalized by dividing them by the relevant seasonal component. Then, to limit the destructive impact of outliers on the forecasts, a squashing function, log(.), is applied. The resulting deseasonalization can be expressed as follows:

$$x_t = \log\left(\frac{y_t}{s_t}\right) \tag{4}$$

where x_t is the deseasonalized t-th element of the normalized TS, and s_t is the t-th seasonal component.

The preprocessed TS sequences contained in the successive input and output windows are represented by vectors: $\mathbf{x}_t^{in} = [x_t \dots x_{t+12}]$, $\mathbf{x}_t^{out} = [x_{t+13} \dots x_{t+24}]$, $t = 1, ..., N-24$. These vectors are included in the training subset for the i-th TS: $\Phi_i = \{(\mathbf{x}_t^{in}, \mathbf{x}_t^{out})\}_{t=1}^{N-24}$. The training subsets for all M TS are combined and form the training set $\Psi = \{\Phi_1, ..., \Phi_M\}$ which is used for RD-LSTM cross-learning. Note the dynamic character of the training set. It is updated in each epoch because the seasonal components in (4) are updated by SGD.

The forecasts produced by RD-LSTM, $\hat{x}_t$, are reseasonalized as follows:

$$\hat{y}_t = s_t \exp(\hat{x}_t) \tag{5}$$

where s_t is determined from (3) on the basis of the TS history.

Finally, the TS is denormalized using transformed equation (2):

$$\hat{z}_{i,j} = (\hat{y}_{i,j} - 1)\hat{\sigma}_i + \hat{\overline{z}}_i \tag{6}$$

where i refers to the forecasted yearly period, $j = 1, ..., 12$, $\hat{\overline{z}}_i$ and $\hat{\sigma}_i$ are the forecasted yearly mean and dispersion for period i.

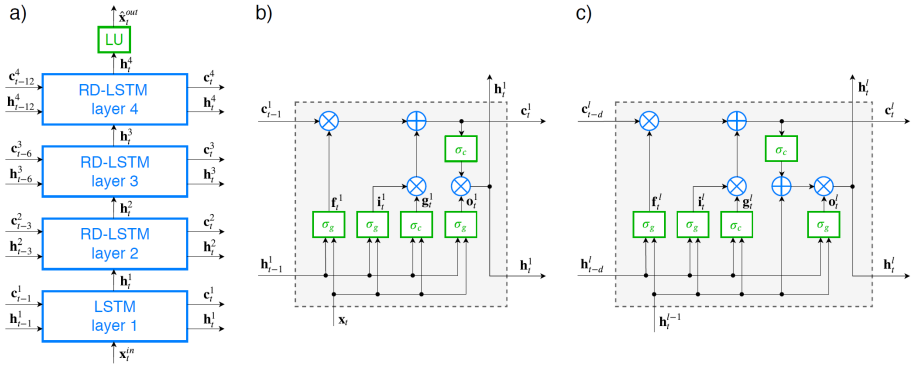

Fig. 3. RD-LSTM architecture (a), LSTM block (b), and RD-LSTM block (c)

Table 1. Equations for the forward pass of LSTM blocks

Standard LSTM block	RD-LSTM block
$\mathbf{f}_t^1 = \sigma_g(\mathbf{W}_f^1\mathbf{x}_t + \mathbf{V}_f^1\mathbf{h}_{t-1}^1 + \mathbf{b}_f^1)$	$\mathbf{f}_t^l = \sigma_g(\mathbf{W}_f^l\mathbf{h}_t^{l-1} + \mathbf{V}_f^l\mathbf{h}_{t-d}^l + \mathbf{b}_f^l)$
$\mathbf{i}_t^1 = \sigma_g(\mathbf{W}_i^1\mathbf{x}_t + \mathbf{V}_i^1\mathbf{h}_{t-1}^1 + \mathbf{b}_i^1)$	$\mathbf{i}_t^l = \sigma_g(\mathbf{W}_i^l\mathbf{h}_t^{l-1} + \mathbf{V}_i^l\mathbf{h}_{t-d}^l + \mathbf{b}_i^l)$
$\mathbf{g}_t^1 = \sigma_c(\mathbf{W}_g^1\mathbf{x}_t + \mathbf{V}_g^1\mathbf{h}_{t-1}^1 + \mathbf{b}_g^1)$	$\mathbf{g}_t^l = \sigma_c(\mathbf{W}_g^l\mathbf{h}_t^{l-1} + \mathbf{V}_g^l\mathbf{h}_{t-d}^l + \mathbf{b}_g^l)$
$\mathbf{o}_t^1 = \sigma_g(\mathbf{W}_o^1\mathbf{x}_t + \mathbf{V}_o^1\mathbf{h}_{t-1}^1 + \mathbf{b}_o^1)$	$\mathbf{o}_t^l = \sigma_g(\mathbf{W}_o^l\mathbf{h}_t^{l-1} + \mathbf{V}_o^l\mathbf{h}_{t-d}^l + \mathbf{b}_o^l)$
$\mathbf{c}_t^1 = \mathbf{f}_t^l \otimes \mathbf{c}_{t-1}^1 + \mathbf{i}_t^1 \otimes \mathbf{g}_t^1$	$\mathbf{c}_t^l = \mathbf{f}_t^l \otimes \mathbf{c}_{t-d}^l + \mathbf{i}_t^l \otimes \mathbf{g}_t^l$
$\mathbf{h}_t^1 = \mathbf{o}_t^1 \otimes \sigma_c(\mathbf{c}_t^1)$	$\mathbf{h}_t^l = \mathbf{o}_t^l \otimes (\sigma_c(\mathbf{c}_t^l) + \mathbf{h}_t^{l-1})$

where $\mathbf{W}$, $\mathbf{V}$ and $\mathbf{b}$ are input weights, recurrent weights and biases, respectively, σ_c is a hyperbolic tangent function, σ_g is a sigmoid activation function $(1 + e^{-x})^{-1}$, $\otimes$ denotes the Hadamard product, superscript 1 refers to the first layer of RD-LSTM network, where we use the standard LSTM block, superscript l indicates the layer number for RD-LSTM blocks (from 2 to 4 in our case), and d is a dilation (3, 6 or 12 in our case).

2.3 Residual Delated LSTM

The RD-LSTM architecture used in this study is shown in Fig. 3(a) [1]. It is composed of four recurrent layers and a linear unit LU. The first layer consists of the standard LSTM block shown in Fig. 3(b). The subsequent three layers consist of RD-LSTM blocks, i.e. blocks equipped with dilated recurrent skip connections and a spatial shortcut path from lower layers (Fig. 3(c)).

A standard LSTM block consists of hidden state $\mathbf{h}_t$ and cell state $\mathbf{c}_t$. The cell state contains information learned from the previous time steps which can be added to or removed from the cell state using the gates: input gate (i), forget gate (f) and output gate (o). At each time step t, the block uses the past state, $\mathbf{c}_{t-1}$ and $\mathbf{h}_{t-1}$, and input $\mathbf{x}_t$ to compute output $\mathbf{h}_t$ and updated cell state $\mathbf{c}_t$. The hidden and cell states are recurrently connected back to the block input. All of the gates are controlled by the hidden state of the past cycle and input $\mathbf{x}_t$. The equations for a standard LSTM block are shown in Tab. 1.

The RD-LSTM blocks employ dilation mechanism proposed in [22]. It is to solve three main problems related to RNN learning on long sequences: complex dependencies, vanishing and exploding gradients, and efficient parallelization. It is characterized by multi-resolution dilated recurrent skip connections. To com-

pute the current states of the LSTM block, the last $d-1$ states are skipped, i.e. a dilated LSTM block receives as input states $\mathbf{c}_{t-d}$ and $\mathbf{h}_{t-d}$, where $d > 1$ is a dilation. Usually multiple dilated recurrent layers are stacked with hierarchical dilations to construct a system, which learns the temporal dependencies of different scales at different layers. In [22], it was shown that this solution can reliably improve the ability of recurrent models to learn long-term dependency. Dilated RNN can be particularly useful for seasonal TS. In this case dilations can be related to seasonality. In our case we use $d = 3$, 6 and 12.

A residual LSTM was proposed in [23] to enable effective training of deep networks with multiple LSTM layers by avoiding vanishing or exploding gradients in the temporal domain. Residual LSTM provides a shortcut path between adjacent layer outputs. The shortcut paths are used to allow gradients to flow through a network directly, without passing through non-linear activation functions. In our implementation, we introduced shortcut paths extending equation for the hidden state (note additional component, $\mathbf{h}_t^{l-1}$, for a hidden state in the right column of Table 1, where the RD-LSTM computation process is shown).

A linear unit, LU, transforms the output of the last layer, $\mathbf{h}_t^4$, into the forecast of the output x-vector:

$$\hat{\mathbf{x}}_t^{out} = \mathbf{W}_x \mathbf{h}_t^4 + \mathbf{b}_x \tag{7}$$

Note that RD-LSTM works on 12-component x-vectors. It produces the forecasts for the whole yearly period receiving the previous yearly period as input. The parameters of RD-LSTM, i.e. input weights $\mathbf{W}$, recurrent weights $\mathbf{V}$, and biases $\mathbf{b}$, are learned using SGD in the cross-learning mode simultaneously with the ETSd parameters. The length of the cell and hidden states, m, the same for all layers, was selected on the training set to ensure the highest performance.

3 Results

The proposed forecasting model is applied for monthly electricity demand forecasting for 35 European countries. The real-world data are taken from the ENTSO-E repository (www.entsoe.eu). The TS lengths vary from 5 to 24 years. The forecasting problem is to produce the forecasts for the twelve months of 2014 (last year of data) using data from the previous period for training. For hyperparameter selection the model learned on the TS fragments up to 2012, and then it was validated on 2013. The selected hyperparameters were used to build the model for 2014: number of epochs 10, learning rate 10^{-3}, length of the cell and hidden states $m = 40$, asymmetry parameter in pinball loss $\tau = 0.4$, ensembling parameters: $L = 5$, $K = 4$, $R = 3$. RD-LSTM was implemented in C++ relying on the DyNet library and run in parallel on an 8-core CPU. We employ R implementation of ETS (function `ets` from package `forecast`).

The proposed model was compared with its predecessor, ETS+RD-LSTM [1], and other state-of-the-art models based on ML as well as classical statistical models. They include: k-nearest neighbor weighted regression model, k-NNw,

fuzzy neighborhood model, FNM, general regression NN model, GRNN, multilayer perceptron, MLP, adaptive neuro-fuzzy inference system, ANFIS, LSTM model, ARIMA model, and ETS model. All ML models were used also in +ETS versions, where the TS were initially normalized and the yearly mean and dispersion were forecasted using ETS (just like in this study). Details of the comparative models can be found in [18, 24, 25, 19].

Table 2 shows the forecast results averaged over 35 countries, i.e. median of absolute percentage error (APE), mean APE (MAPE), interquartile range of APE as a measure of the forecast dispersion, root mean square error (RMSE), and mean PE (MPE). The proposed model is denoted by 3ETS+RD-LSTM. As can be seen from this table, all error measures indicate that 3ETS+RD-LSTM is the most accurate model comparing with its competitors. It outperforms its predecessor, ETS+RD-LSTM, by 8.7% in MAPE and 9.5% in RMSE.

Table 2. Results comparison among proposed and comparative models

Model	Median *APE*	*MAPE*	*IQR*	*RMSE*	*MPE*
k-NNw	2.89	4.99	3.85	368.79	-1.87
FNM	2.88	4.88	4.26	354.33	-2.03
N-WE	2.84	5.00	3.97	352.01	-1.91
GRNN	2.87	5.01	4.02	350.61	-1.87
k-NNw+ETS	2.71	4.47	3.52	327.94	-1.25
FNM+ETS	2.64	4.40	3.46	321.98	-1.26
N-WE+ETS	2.68	4.37	3.36	320.51	-1.26
GRNN+ETS	2.64	4.38	3.51	324.91	-1.26
MLP	2.97	5.27	3.84	378.81	-1.37
MLP+ETS	3.11	4.80	4.12	358.07	-1.71
ANFIS	3.56	6.18	4.87	488.75	-2.51
ANFIS+ETS	3.54	6.32	4.26	464.29	-1.30
LSTM	3.73	6.11	4.50	431.83	-3.12
LSTM+ETS	3.08	5.19	4.54	366.45	-1.41
ARIMA	3.32	5.65	5.24	463.07	-2.35
ETS	3.50	5.05	4.80	374.52	-1.04
ETS+RD-LSTM	2.74	4.48	3.55	347.24	-1.11
3ETS+RD-LSTM	2.64	4.09	3.13	314.01	-0.32

MPE provides information on potential forecast bias. All the models produced negatively biased forecasts, i.e. overpredicted. But for 3ETS+RD-LSTM, the t-test did not reject the null hypothesis that PE comes from a normal distribution with mean equal to zero (p-value = 0.44). All other models did not pass this test. So it can be concluded that 3ETS+RD-LSTM, as the only model, produced unbiased forecasts. Note that 3ETS+RD-LSTM has the mechanism to deal with bias. The loss function (1) asymmetry is controlled by parameter τ. It was selected as 0.4, which allowed the model to reduce the negative bias.

Fig. 4 depicts more detailed results, MAPE for each country. As can be seen, in most cases 3ETS+RD-LSTM is one of the most accurate models. Fig. 5 depicts

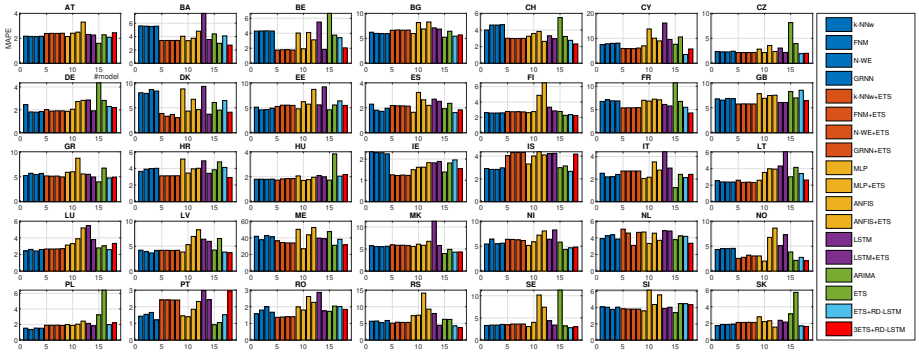

Fig. 4. MAPE for each country

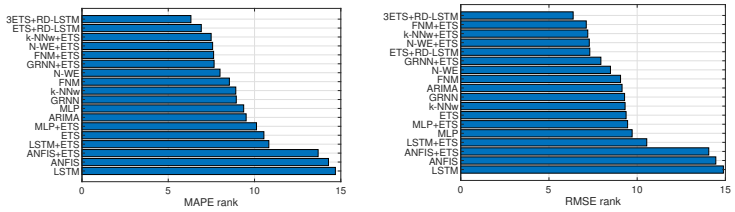

Fig. 5. Rankings of the models

the model rankings based on MAPE and RMSE. They show average ranks of the models in the rankings for individual countries. Note the first position of 3ETS+RD-LSTM in both rankings.

Examples of forecasts produced by the selected models are shown in Fig. 6. For PL and DE data, MAPE is on the low level around 2% while for GB data the forecasts are strongly underestimated, over 5%. This is because the demand for GB went up unexpectedly in 2014 despite the downward trend observed in the previous period.

Summarizing experimental research, it should be noted that the forecasting model performance depends significantly on the appropriate TS preprocessing. Although LSTM deals with raw data, without preprocessing [19], introducing initial normalization and dynamic deseasonalization in 3ETS+RD-LSTM improved significantly LSTM performance.

It should be noted that LSTM based models are more complex than other comparative models. Due to the huge number of parameters and complicated learning procedure using backpropagation through time, the learning time of LSTM is much longer than for other comparative models.

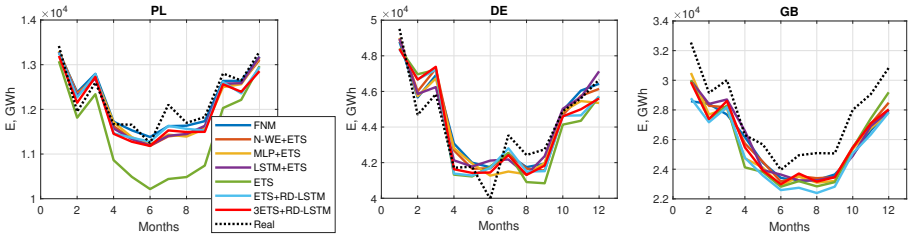

Fig. 6. Examples of forecasts produced by selected models

4 Conclusion

In this work, we proposed an extended hybrid RD-LSTM and ETS model for MTLF. It combines initial TS decomposition into three components (normalized TS, trend, and dispersion), ETS modules for trend and dispersion forecasting, ETS for deseasonalization, advanced LSTM, and ensembling. The model has a hierarchical structure composed of a global part learned across many TS (LSTM) and a TS specific part (normalization and deseasonalization). Deseasonalization and LSTM are combined in a simultaneous learning process using SGD which leads to learning TS representations and mapping at the same time.

We used residual dilated LSTM, which can capture better long-term seasonal relationships and ensure more efficient training. This is because of dilated recurrent skip connections and a spatial shortcut path from lower layers. To deal with a forecast bias, an asymmetric pinball loss function was applied. Three-level ensembling provides regularization reducing the model variance, which has sources in the stochastic nature of SGD, and also in data and parameter uncertainty.

An experimental study, monthly electricity demand forecasting for 35 European countries, demonstrated the state-of-the-art performance of the proposed model. It generated more accurate forecasts than its predecessor (ETS+RD-LSTM), classical models such as ARIMA and ETS as well as state-of-the-art models based on ML.

References

1. Dudek, G., Pełka, P., Smyl, S.: A Hybrid Residual Dilated LSTM end Exponential Smoothing Model for Mid-Term Electric Load Forecasting. ArXiv:2004.00508 (2020)
2. Suganthi, L., Samuel, A.-A.: Energy models for demand forecasting — A review. RENEW SUST ENERG REV **16**(2), 1223–1240 (2002)
3. Barakat, E.H.: Modeling of nonstationary time-series data. Part II. Dynamic periodic trends. INT J ELEC POWER **23**, 63–68 (2001)
4. González-Romera, E., Jaramillo-Morán, M.-A, Carmona-Fernández, D.: Monthly electric energy demand forecasting with neural networks and Fourier series. ENERG CONVERS MANAGE **49**, 3135–3142 (2008)
5. Chen, J.F., Lo, S.K., Do, Q.H.: Forecasting monthly electricity demands: An application of neural networks trained by heuristic algorithms. Information **8**(1), 31 (2017)

6. Gavrilas, M, Ciutea, I, Tanasa, C.: Medium-term load forecasting with artificial neural network models. IEEE Conf. Elec. Dist. Pub. **6** (2001)
7. Doveh, E, Feigin, P., Hyams, L.: Experience with FNN models for medium term power demand predictions. IEEE Trans. Power Syst. **14**(2), 538–546 (1999)
8. Pełka, P., Dudek, G.: Medium-term electric energy demand forecasting using generalized regression neural network. In: Information Systems Architecture and Technology ISAT 2018, AISC, vol. 853, pp. 218–227, Springer, Cham, (2018)
9. Pei-Chann, C., Chin-Yuan, F., Jyun-Jie, L.: Monthly electricity demand forecasting based on a weighted evolving fuzzy neural network approach. INT J ELEC POWER **33**, 17–27 (2011)
10. Ahmad, T., Chen, H.: Potential of three variant machine-learning models for forecasting district level medium-term and long-term energy demand in smart grid environment. Energy **160**, 1008–1020 (2018)
11. Hewamalage, H., Bergmeir, C., Bandara, K.: Recurrent neural networks for time series forecasting: Current status and future directions. arXiv:1909.00590v3 (2019)
12. Yan, K., Wang, X., Du, Y., Jin, N., Huang, H., Zhou, H.: Multi-Step Short-Term Power Consumption Forecasting with a Hybrid Deep Learning Strategy. Energies **11**(11), 3089 (2018)
13. Bedi, J. Toshniwal, D.: Empirical mode decomposition based deep learning for electricity demand forecasting. IEEE Access **6**, 49144–49156 (2018)
14. Zheng, H., Yuan, J., Chen, L.: Short-Term Load Forecasting Using EMD-LSTM Neural Networks with a XGBboost Algorithm for Feature Importance Evaluation. Energies **10**(8), 1168, (2017)
15. Narayan, A., Hipel, K.-W.: Long short term memory networks for short-term electric load forecasting. Conf Proc IEEE Int Conf Syst Man Cybern (SMC), pp. 2573–2578, (2017)
16. Makridakis, S., Spiliotis, E., Assimakopoulos, V.: The M4 Competition: Results, findings, conclusion and way forward. Int. J. Forecast. **34**(4), 802–808 (2018)
17. Smyl, S.: A hybrid method of exponential smoothing and recurrent neural networks for time series forecasting. Int. J. Forecast. **36**(1), 75–85 (2020)
18. Dudek, G., Pełka, P.: Pattern Similarity-based Machine Learning Methods for Mid-term Load Forecasting: A Comparative Study. ArXiv:2003.01475 (2020)
19. Pełka P., Dudek G.: Pattern-based Long Short-term Memory for Mid-term Electrical Load Forecasting. IJCNN 2020, ArXiv:2004.11834 (2020)
20. Hyndman, R.-J., Koehler, A.-B., Ord, J.-K., Snyder, R.-D, Forecasting with Exponential Smoothing: The State Space Approach. Springer (2008)
21. Oreshkin, B.-N., Carpov, D., Chapados, N., Bengio, Y.: N-BEATS: Neural basis expansion analysis for interpretable time series forecasting. arXiv:1905.10437v4 (2020)
22. Chang, S., Zhang, Y., Han, W., Yu, M., Guo, X., Tan, W., et al.: Dilated recurrent neural networks. arXiv: 1710.02224 (2017)
23. Kim, J., El-Khamy, M., Lee, J.: Residual LSTM: Design of a deep recurrent architecture for distant speech recognition. arXiv:1701.03360 (2017)
24. Pełka, P., Dudek, G.: Pattern-based forecasting monthly electricity demand using multilayer perceptron. In: Artificial Intelligence and Soft Computing ICAISC 2019, LNCS, vol. 11508, pp. 663–672, Springer, Cham (2019).
25. Pełka, P., Dudek, G.: Neuro-Fuzzy System for Medium-term Electric Energy Demand Forecastig. In: Information Systems Architecture and Technology ISAT 2017, AISC, vol. 655, pp. 38–47, Springer, Cham (2018).